

An examination of the security of the TR-31 and AS 2805 schemes

Arthur Van Der Merwe *

David Paul

Jelena Schmalz

Timothy M. Schaerf

School of Science and Technology

University of New England

Armidale

NSW, 2351

Australia

Abstract

In our review of payments cryptography, we analyze the security of the Australian card payment system. In the analysis, we examine existing cryptographic protocols using well-known techniques. We discuss the Australian Triple-DES variant methods (AS 2805) which offer key usage division. We then compare Triple-DES variants with alternative mechanisms to discover a significant weakness in the AS 2805 methods.

Subject Classification: 64A60.

Keywords: AS 2805, Authenticated encryption, Financial, Encryption, TR-31.

1. Introduction

The protection of the financial system in every country is one of the primary directives of payment system regulators. Vulnerable architectures in the financial system might cause uncertainty in the operational effectiveness of economic activities. Historically, each country relied on a combination of international and local standards to maintain payment systems' efficacy and security. In 2006, card manufacturing organisations

This research is supported by an Australian Government Research Training Program (RTP) Scholarship.

* E-mail: avande22@myune.edu.au

started a commercial company to standardise security practices across the globe. The commercial company, named Payment Card Industry Security Standards Council (PCI-SSC), headed by US-based card companies (Visa, Mastercard and American Express), imposes security requirements and specifications. The PCI-SSC produce industry specifications which are adopted widely in several regions [12]. The PCI-SSC industry standards are a collection of security and test requirements. Authorised laboratories examine if participants meet the desired specifications based on their role in the payments landscape. Recently, PCI-SSC mandated the management of all cryptographic keys in structures called “Key blocks”. The ANSI X9 TR-31 Technical Report (TR-31) [2] defines Key blocks, which was in reply to the ANSI X9 24.1 Standard [1]. The goal of ANSI’s X9.24-1-2017 is to stipulate the minimum requirements for managing symmetric keys in financial transactions. The ANSI requirements apply to POS and ATM transactions, and by definition, to banking messages.

2. Our Contribution

Since the development of the Australian Financial Transaction System (AS 2805) series of standards by the Australian Standards IT-005 committee, an open problem remained to describe the security of AS 2805. TR-31 has the same open question, though TR-31 is an authenticated encryption scheme. We investigate TR-31 and AS 2805 then draw a parallel by reviewing the distinctions and relative advantages between the two systems. We use standard methods and existing models to demonstrate the efficiency of breaking AS 2805 by studying established literature and its applicability. In our investigation, we look at two distinct methods of integrity. Plaintext integrity INT-PTXT necessitates that producing a ciphertext that can be decrypted to a message, the scheme has never encrypted is computationally infeasible. Ciphertext integrity INT-CTXT, on the other hand, requires that producing a ciphertext not previously produced by the scheme be computationally infeasible, regardless of whether the underlying plaintext is novel. For privacy of the schemes, in our analysis, we look at indistinguishability against chosen-plaintext attacks (IND-CPA) and indistinguishability against chosen-ciphertext attacks (IND-CCA), whose security is considered based on a randomised scheme. While for non-malleable chosen-plaintext attacks (NM-CPA), it is infeasible for a modified ciphertext of a scheme to decrypt to a related plaintext. We present our results in Table [2.1] .

Table 2.1
Financial encryption composition security.

Scheme	Privacy			Integrity	
	IND-CPA	IND-CCA	NM-CPA	INT-PTXT	INT-CTXT
TR-31	secure	secure	secure	secure	secure
AS 2805	insecure	insecure	insecure	insecure	insecure

3. Preliminaries

We rely on some formal aspects and other self-defined notions. We abstract the use of any specific encryption scheme and rely on general constructions. We start by giving an overview of the tools and definitions required for our review and results. We establish the notation for block ciphers, experiments and message authentication. The study assumes some prior knowledge of encryption schemes and operations.

3.1 Notation

We use $[n]$ for a positive integer n to denote the set $\{1, \dots, n\}$ and $\emptyset$ for an empty set. A binary string of size n is represented as $\{0, 1\}^n$ and $\{0, 1\}^*$ represents a set of all strings of finite length. For any two binary strings s_1 and s_2 , we denote the size of s_1 as $|s_1|$ and the concatenation of the string s_1 immediately followed by s_2 as $s_1 \| s_2$. For any non-negative integer $z \leq |s_1|$, we use $\lfloor s_1 \rfloor_z$ to represent the string obtained by the truncation of s_1 to the leftmost z bits. The process of uniformly sampling a value from a finite set $\mathcal{S}$ and assigning the result to $x \in \mathcal{S}$ is written as $x \xleftarrow{\$} \mathcal{S}$ and $\mathcal{R}$ denotes a randomised space. We use the $\oplus$ symbol to denote the binary addition modulo two, or exclusive OR, (XOR) operator and $\perp$ to indicate an error.

Our model security is utilising the code-based game-playing framework by Bellare et. al. [4] where the interaction between the adversary with the game is implicit. Throughout the text, we refer to a single-use arbitrary number in cryptographic algorithms as a *nonce*. In practice a *nonce* is often a pseudo-random number which introduces randomness into an algorithm to prevent replay attacks. A starting variable, also known as an initialisation vector, is an input to a cryptographic primitive that is used to establish the initial state. The IV must be random or pseudorandom in most cases, but it can also be unpredictable or unique in some cases.

AS 2805 Payments Terminology

The AS 2805 series of standards depicts session keys that encrypt financial data elements as variants. That is, given a key $k \xleftarrow{\$} \mathcal{K}$, we term a master key, from some key space $\mathcal{K}$, the system applies a function $k_v \leftarrow f(v, k)$ to derive a variant key k_v , also called a session key. The function $f(\cdot)$ is considered a standard key derivation function, e.g, HKDF [11]. In the case of AS 2805, the value v is public, and the function f is the $\oplus$ operation. The AS 2805 system may generate a single random master key applying $f(\cdot)$ which derives several variant keys, each time changing the variant value v . The value of v is bound to the purpose of the key.

Random Experiments

We refer to a protocol game played by a group of interactive probabilistic algorithms as random experiments. We express random experiments as a list of actions by players, where the result of the actions is an event with a discrete probability, denoted as:

$$\Pr[\text{action}_1; \text{action}_2; \dots; \text{action}_n : \text{event}]. \quad (1)$$

The outcome of the game in [1] is the probability of event after executing $\text{action}_1; \dots; \text{action}_n$ in sequential order. The event is taken over a probability space, of all random variables involved in the actions.

3.2 Block Ciphers

A block cipher [5] is a deterministic cipher $\mathcal{E}$ with an encrypt $E(\cdot)$ and decrypt $D(\cdot)$ function defined over a message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$. The message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ are in the same finite set $\mathcal{X}$, $\mathcal{M} \times \mathcal{C} \in \mathcal{X}$. We call an element $x \in \mathcal{X}$ a data block, and refer to $\mathcal{X}$ as the data block space of $\mathcal{E}$. For every fixed key $k \in \mathcal{K}$, we can define the function $f_k := E_k(\cdot)$; that is, $f_k, \mathcal{X} \rightarrow \mathcal{X}$ sends $x \in \mathcal{X}$ to $E_k(x) \in \mathcal{X}$. The usual correctness requirement for any cipher implies that, for every fixed key k , the function f_k is one-to-one and, as $\mathcal{X}$ is finite, f_k must be onto as well. Thus, f_k is a permutation on $\mathcal{X}$, and $D_k(\cdot)$ is the inverse permutation f_k^{-1} .

Triple DES

This article refers to the Triple-DES block cipher as TDES, while it is officially known as the Triple Data Encryption Algorithm (TDEA or Triple DEA) [6]. TDES is a secure encryption cipher widely used in financial services. The TDES cipher is the combination of three independent DES operations, with either two or three distinct keys. We define the three key TDES encryption and decryption operations, using keys k_1 , k_2 and k_3 and plaintext m producing a ciphertext c as follows: $c \leftarrow E_{k_3}(D_{k_2}(E_{k_1}(m)))$, $m \leftarrow D_{k_1}(E_{k_2}(D_{k_3}(c)))$. In two key TDES we have $c \leftarrow E_{k_1}(D_{k_2}(E_{k_1}(m)))$, $m \leftarrow D_{k_1}(E_{k_2}(D_{k_1}(c)))$.

3.3 Message Authentication Code (MAC)

A message integrity system based on a shared secret key is called a Message Authentication Code or MAC for short. We limit our review and discussion to MAC systems that use symmetric keys as defined by Boneh et. al. [5].

Definition 1 (Message Authentication [5]) : A MAC system $\mathcal{I} = (\text{Mac}, \text{Verify})$ is a pair of *efficient* algorithms, Mac and Verify, where Mac is called a signing algorithm and Verify is called a verification algorithm. Algorithm Mac is used to generate tags and algorithm Verify is used to verify tags.

- Mac is a probabilistic algorithm that is invoked as $\tau \leftarrow \text{Mac}_k(m)$, where k is a key, m is a message, and the output τ is called a tag.
- Verify is a deterministic algorithm that is invoked as $r \leftarrow \text{Verify}_k(m, \tau)$, where k is a key, m is a message, τ is a tag, and the output r is either “accept” or “reject”.
- We require that tags generated by Mac_k are always accepted by Verify_k ; that is, the MAC system must satisfy a correctness property, such that for every valid key k and message pair, we have $\text{Verify}_k(m, \text{Mac}_k(m)) = \text{accept}$.

$\mathcal{I} = (\text{Mac}, \text{Verify})$ is defined over $(\mathcal{K}, \mathcal{M}, \text{Mac})$. Whenever algorithm Verify outputs “accept” for some message-tag pair (m, τ) , we say that τ is a valid tag for m under key k , or that (m, τ) is a valid pair under k . The simplest type of system is one in which the signing algorithm Mac is deterministic, and the verification algorithm is defined as:

$$\text{Verify}_k(m, \tau) = \begin{cases} \text{accept} & \text{if } \text{Mac}_k(m) = \tau, \\ \text{reject} & \text{otherwise.} \end{cases} \quad (2)$$

We call the MAC system in [2] a deterministic MAC system. A deterministic MAC system has unique tags for each key, such that: for a given key k , and a given message m , there is a unique valid tag for m under k .

4. AS 2805 Key Management

In the AS 2805 series of standards [7–10], the AS 2805.6.x range covers key management. Specifically, the vast majority of payment devices deployed in Australia use four standards for the management of symmetric keys:

- AS 2805.6.2 – Terminal to host key management using transaction keys.
- AS 2805.6.3 – Node to node management using session keys.
- AS 2805.6.4 – Terminal to host key management using session keys.
- AS 2805.9 – Privacy of communications.

The AS 2805.6.1 (Section 9.3.2.2) standard outlines an algorithm for inferring session keys, where the standard establishes the notion of variant keys. AS 2805.6.4 [10] use a series of public constants. The AS 2805.6.4 [10] and AS 2805.6.3 standards re-affirm variants using TDES CBC mode and a zero string initialisation vector.

Notes On Variants

As defined in the AS 2805 standards, the variants method is intended to separate different keys for various purposes. Unfortunately, the AS 2805 use of variants possesses vulnerabilities. Furthermore, the variant calculation functions are invertible and public. Knowledge of a master key would enable an adversary to recover all variant keys by applying the variant function with the appropriate variant. The recovery of a variant key further allows an attacker to change from one variant to another. The variant keys in AS 2805.6.4 [10] are session keys, which encrypt data and are used in the generation of message authentication tags.

5. ANSI X9 TR-31 Key Management

The 2001 TR-31 report [2] describes a compliant mechanism that conforms to the requirements in the ANSI X9.24.1 standard [1] to bind key purpose information with the key. The management of distinct keys for each type of operation is comparable to variants. However, the TR-31 mechanism is very different; it has clear text header information that describes the purpose, instead of variant bits encrypted with the key. Additionally, another key encrypts the data, and then a MAC is generated on the encrypted data, including the header information. Similar to AS 2805, these encrypted keys are transmitted between parties or stored.

6. Comparison of AS 2805 and TR-31

We now analyse the two financial schemes, AS 2805 and TR-31. To do this, we take motivation from the composition of privacy and integrity detailed in [3]. The TR-31 scheme corresponds to MAC-then-Encrypt (MtE), whereas AS 2805 conforms to Encrypt-then-MAC (EtM). The work of [3] asserts that EtM works well and is alone secure when the underlying primitives are sound. That is, when an IND-CCA secure scheme is combined with a SUF-CMA secure MAC, we could have a secure composition. Our analysis of AS 2805 focuses on messages in transit. Transit messages have a CBC MAC calculated on a variable-length payment message. AS 2805 messages are very different from TR-31. TR-31 focuses on cryptographic keys and produces a MAC on each key, not the message. The AS 2805 scheme does not have a MAC on individual cryptographic keys, but rather the entire message. Therefore, we assume a single key message, with the message elements, not including the cryptographic key, as the associated data.

6.1 AS 2805 (EtM) Analysis

The AS 2805 Encrypt-then-MAC (EtM) scheme combines a symmetric encryption algorithm $\mathcal{E} = E(k_e, k_m, m, p)$ and a symmetric decryption algorithm $D(k_e, k_m, c)$, outlined in Table [6.1] that has additional plaintext p , that is not encrypted and a zero IV using the following algorithm, given a uniformly sampled encryption key: $k_e \xleftarrow{\$} \mathcal{K}$, a message authentication key $k_m \xleftarrow{\$} \mathcal{K}$ and IV $\leftarrow 0^{|k_e|}$.

Table 6.1
AS 2805 (EtM) algorithm.

	Algorithm $E(k_e, k_m, m, p)$	Algorithm $D(k_e, k_m, c)$
1	$c \xleftarrow{\$} E_{k_e}(m, IV)$	$p, c, \tau \leftarrow d$
2	$\tau \xleftarrow{\$} \text{Mac}_{k_m}(c \parallel p)$	$v \leftarrow \text{Verify}_{k_m}(c, \tau)$
3	$d \leftarrow p \parallel c \parallel \tau$	$m \leftarrow D_{k_e}(c)$
4	return d	If $v = \text{true}$ then return m
5		else return $\perp$

AS 2805 uses a fixed IV in CBC encryption, making the encryption scheme deterministic. In our analysis, we can see that AS 2805 is not IND-CPA, IND-CCA or NM-CPA secure. The malleability of CBC mode in the deterministic setting leads us to determine that an adversary is able to find a corresponding ciphertext with a binary relation to the plaintext, violating privacy. CBC MAC is only secure if you restrict the MAC to strings in the domain $\{0,1\}^m$ for some positive integer m [3]. If a CBC MAC is applied to a string varying in length, then an adversary can distinguish the object from a random function. This attack is possible because, when given a tag on a message $\tau \leftarrow \text{Mac}_k(m_1)$, one can XOR the tag τ with a second message $m'_2 \leftarrow m_2 \oplus \tau$ and compute the tag on $\tau' \leftarrow \text{Mac}_k(m'_2)$. It turns out that τ' is a valid tag for both m_1 and m_2 . We capture this advantage in the following LR security game:

Adversary: $\mathcal{A}^{\text{Mac}_k(\cdot, b)}$

Game: $\text{Exp}_{\text{CBC-MAC}}^b$

$m_1 \leftarrow 1^n; m_2 \xleftarrow{\$} \mathcal{R}$

$\tau_1, m_1 \leftarrow \text{Mac}_k(m_1, b)$

$\tau_2, m_2 \leftarrow \text{Mac}_k(m_2, b)$

$\tau_3 \leftarrow m_2 \oplus \tau_1$

$m_3 \leftarrow m_1 \parallel \tau_3 \parallel m_2$

$\tau_3, m_3 \leftarrow \text{Mac}_k(m_3, b)$

If $\tau_3 = \tau_2$, then return 1 else return 0;

Then we have:

$$\text{Adv}_S^{\text{CBC-MAC}}(\mathcal{A}^{\text{Mac}_k(\cdot, b)}) = \Pr[\text{Exp}_{\text{CBC-MAC}}^b(\mathcal{A}^{\text{Mac}_k(\cdot, b)})] = 1.$$

This problem cannot be solved by adding a message-size block. We recommend the use of CMAC (like in TR-31) for variable length message, to mitigate against this attack. In comparing our ideal cipher to the AS 2805 construction, we note that an adversary can attack the $E(k_e, k_m, m, p)$ and $D(k_e \parallel k_m, c)$ algorithm to extract keying information.

6.2 TR-31 Mac-then-Encrypt (MtE) analysis

The TR-31 Mac-then-Encrypt (MtE) scheme combines a symmetric encryption algorithm $\mathcal{E} = E(k_e, k_m, m, p)$ and a symmetric decryption algorithm $\mathcal{D} = D(k_e, k_m, d)$, where τ output of the MAC is the IV of the symmetric scheme and the MAC is computed over additional plaintext p , that is not encrypted with the following algorithm, given a uniformly selected encryption key $k_e \xleftarrow{\$} \mathcal{K}$ and a message authentication key $k_m \xleftarrow{\$} \mathcal{K}$. We describe this algorithm in Table 6.2.

Table 6.2
TR-31 (MtE) algorithm.

	Algorithm $E(k_e, k_m, m, p)$	Algorithm $D(k_e, k_m, d)$
1	$\tau \xleftarrow{\$} \text{Mac}_{k_m}(p \parallel m)$	$c, \tau, p \leftarrow d$
2	$c \xleftarrow{\$} E_{k_e}(m, \tau)$	$m \leftarrow D_{k_e}(c, \tau)$
3	$d \leftarrow p \parallel c \parallel \tau$	$v \leftarrow \text{Verify}_{k_m}(p \parallel m, \tau)$
4		If $v = \text{true}$ then return m
5		else return $\perp$

The tag τ of the MAC is computed on a fixed size message, outputting randomised data which is then used as the IV of the encryption scheme. The encryption is probabilistic in nature, and secure under IND-CPA, IND-CCA, NM-CPA. We summarise our final results in Table [2.1].

7. Conclusion

We investigated the security aspects of the AS 2805 and TR-31 schemes in our work. We identified that the AS 2805 scheme has apparent differences from TR-31 and is vulnerable to various textbook attacks.

Arthur van der Merwe : 0000-0001-9146-5433

David Paul : 0000-0002-2428-5667

Jelena Schmalz : 0000-0001-5486-7679

Timothy M. Schaerf : 0000-0001-6642-8374

References

- [1] ANSI: Ansi x9.24-1-2017: Retail financial services symmetric key management part 1: Using symmetric techniques. Tech. rep., American National Standards Institute (2017).
- [2] ANSI: Ansi x9 tr 31-2018: Interoperable secure key exchange key block specification. Tech. rep., American National Standards Institute (2018).
- [3] Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *Journal of Cryptology* 21(4), 469–491 (2008).
- [4] Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 409–426. Springer (2006).
- [5] Boneh, D., Shoup, V.: A graduate course in applied cryptography. Draft 0.2 (2015).
- [6] ISO Central Secretary: Information technology - Security techniques – Encryption algorithms- Part 3: Block ciphers. Standard ISO/IEC 18033.3:2020, International Organization for Standardization, Geneva, CH (2012), <https://www.iso.org/standard/54531.html>.
- [7] IT-005, Financial Transaction Systems: Electronic funds transfer – Requirements for interfaces, Part 6.3: Key management - Session keys - Node to node. Standard AS 2805.6.3:2000, Standards Australia, Sydney, AU (2000).
- [8] IT-005, Financial Transaction Systems: Electronic funds transfer – Requirements for interfaces, Part 9: Privacy of communications. Standard AS 2805.9:2000, Standards Australia, Sydney, AU (2000).
- [9] IT-005, Financial Transaction Systems: Electronic funds transfer – Requirements for interfaces - Key management - Transaction keys. Standard AS 2805.6.2:2002, Standards Australia, Sydney, AU (2002).

- [10] IT-005, Financial Transaction Systems: Electronic funds transfer – Requirements for interfaces - Key management - Session keys - Terminal to acquirer. Standard AS 2805.6.4:2006, Standards Australia, Sydney, AU (2006).
- [11] Krawczyk, H.: Cryptographic extraction and key derivation: The hkdf scheme. In: Annual Cryptology Conference. pp. 631–648. Springer (2010).
- [12] Morse, E.A., Raval, V.: Pci dss: Payment card industry data security standards in context. Computer Law & Security Review 24(6), 540–554 (2008).

Received December, 2021

Revised June, 2022